## D. REMARKS

### Status of the Claims

Claims 1-25 are currently present in the Application, and claims 1, 8, 15, and 22-25 are independent claims. Claims 1-23 stand rejected. No claims have been amended, cancelled, or added in this Response.

### Allowable Subject Matter

Claims 24 and 25 are independent claims and were not rejected in the Office Action. However, the Examiner did not specifically note that these claims were allowable. Applicants respectfully request clarification of the allowability of these claims in the next Office Communication.

### Drawings

Applicant notes with appreciation the Examiner's acceptance of Applicant's formal drawings filed with the application.

### Claim Rejections - Alleged Anticipation Under 35 U.S.C. § 102

Claims 1-23 stand rejected under 35 U.S.C. § 102(e) as allegedly being anticipated by U.S. Patent No. 6,496,793 to Veditz et al. (hereinafter "Veditz"). Applicants respectfully traverse the rejections. Further, as noted above, claims 24 and 25 were not rejected in the Office Action and are assumed to be allowable. This assumption is especially strong since, as discussed below, Veditz does not teach or suggest the limitations set forth in these claims.

In order to anticipate under 35 U.S.C. § 102, the Veditz reference must teach each and every element as set forth in Applicants' claims. MPEP § 2131, citing *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). As discussed below, it is readily apparent that Veditz falls well short of this requirement.

In Applicant's independent claims 1, 8, and 15, Applicant claims a method, information handling system, and computer program product that each include limitations of:

- locating one or more display strings within a management definition data file;
- copying each of the display strings from the management definition data file;
- determining an identifier for each of the display strings;
- writing the identifiers and the corresponding display strings to a translation template; and
- writing the identifier to the management definition data file.

The Office Action contends that Veditz teaches each of these limitations. As an initial matter, Applicants note that Veditz does not teach or suggest anything to do with a "management definition data file," as claimed by Applicants. A management definition file is described in various places in Applicants' specification and is also a term known by those skilled in the art. Namely, a management definition file is used to store a CIM model. CIM stands for a "Common Information Model" which is stored in a management definition (MOF) file and is used to describe a computer information system in a way that is not bound to a particular implementation or platform. Both of these terms and their interrelationship are described in detail in Applicants' Background section of the specification. In addition, both of these terms are well known by those skilled in the art. What is not found in the art, and more specifically what is not found in Veditz, is Applicants' innovation of using the management definition file to store national language information.

Specifically, in claims 1, 8, and 15, display strings (i.e., English language strings), are (1) located in the management definition data file, (2) copied for analysis [copying each …], (3) have an identifier that corresponds to the display strings [determining an identifier…], (4) the identifiers and display strings are written to a translation template [writing the identifiers…], and (5) the identifier is written to the management definition object [writing the identifier…]. In this manner, the management object can be created as traditionally performed in the past using UML and CIM to store the computer information system in a management definition data file (e.g., a MOF file). Applicants' claimed invention can then be used to substitute display strings found in the management definition data file (e.g., English strings) with identifiers that are used for translations. Translators can then translate the display strings to the various languages that are being supported. These translations are included in runtime files (see Claim 2). Now, when a non-English user uses the management definition data file (e.g., using a standard console, such as the Microsoft Management Console (MMC), the Tivoli Console by IBM, the AS/400 system console, and the AIX™ system console (WebSM™), the display strings will be displayed in the user's national language (e.g., Spanish) rather than the language that was originally used to create the management definition data file.

Applicants have provided the summary, set forth above, of Applicants' claimed invention in order for the Examiner to better understand Applicants' claimed invention. Aided with a better understanding of Applicants' claimed invention, Applicants trust that the Examiner will appreciate the stark differences between Applicants' claimed invention an the teaching of Veditz. In furtherance of this understanding, set forth below, Applicants contrast the specific limitations claimed in independent claims 1, 8, and 15 with the teachings of Veditz.

First, the Office Action contends that Veditz teaches Applicants' claimed limitation of "locating one or more display strings within a management definition data file," citing Veditz at col. 16, lines 26-45. However, a review of the cited section of Veditz reveals that Veditz never teaches or suggests anything to do with management definition data files. Instead, Veditz teaches a system that imbeds Language Driver Identifiers (LDIDs) to record the particular language that was used when a given data object was created or modified (see Veditz, abstract). Moreover, in the section cited in the Office Action, Veditz does not teach or suggest "locating" anything from a management definition data file:

> The following example will illustrate application of the principles of the present invention for the operation of opening a file, such as a database file. Referring now to FIGS. 3A-B, a preferred method 300 of the present invention for processing a request to open a file in a system having National Language Support includes the following steps. At step 301, a request is received by the system for opening a file. For example, in the instance of a database application, an open or use (e.g., dBASE USE) command may be issued for opening an existing database file. As is known in the art, a request to open or otherwise obtain a handle to a disk file is typically done in conjunction with a particular access mode, that is, a file can be opened in different ways. For instance, a file may be opened for "read-only" access. In the instance where one needs to both read to and write from a file, a "read/write" access mode or type is appropriate. As

> still yet another type of access, one may need to only append information to an existing file (i.e., write new information to the terminal portion of that file); "append" access may be treated as if the existing data is read-only. Access mode is important as it determines the ability of the system to touch (create/modify) the data object.

The section of Veditz cited above, and Figure 3 of Veditz, teach a method that can retrieve a language driver identifier (LDID) from a file. The section of Veditz cited above does not teach or suggest "locating … display strings within a management definition data file," as claimed by Applicants. Instead Veditz teaches the opening of a file, such as a database, and different access modes ("read only," "append," etc.) in which the file can be opened. Nowhere in this section does Veditz teach or suggest locating any display strings within any such file. Moreover, as previously discussed, nowhere does Veditz teach or suggest retrieving ***anything*** from a management definition data file.

The Office Action contends that Applicants' next limitation, "copying each of the display strings from the management definition data file," is taught by Veditz, citing Figure 2, element 201 being retrieved from a database. Applicants note that Veditz does not have a Figure 2, and instead has Figures 2A-2C. Applicants further note that element 201 could not be located on any of these figures.

The Office Action next contends that Applicants' limitation of "determining an identifier for each of the display strings" is taught by Veditz citing col. 16, lines 49-67, step 303 of Figure 3A, and col. 18, lines 10-67. First, as explained above, Veditz does not identify any "display strings" as claimed by Applicants. Instead, Veditz teaches identifying Language Driver Identifiers (LDIDs), which are not "display strings" as they are not displayed to a user. It follows, therefore, that because

Veditz does not teach or suggest identifying "display strings," Veditz also does not teach or suggest determining identifiers for such display strings. Indeed, step 303 of Figure 3A is labeled and described as "Check LDID in Data File (e.g., Stored in Header)." Veditz's LDID is clearly not a "display string" nor is Veditz's LDID an identifier corresponding to a "display string." Instead, as described by Veditz, as "A descriptor or Language Driver Identifier (LDID) (e.g., in the form of a system-comparable unit) is employed for storing in desired location(s) of a data object information specifying the language driver that was in use when the data object was created or modified. The LDID, which may be in the form of an ID byte, references a set of language driver values (e.g., lookup table of locales). This allows the system of the present invention to intelligently process data objects created or modified under one language driver with those created or modified by a different language driver. In the event of incompatibilities, the system provides error handling routines, including facilities for warning users of incompatible or otherwise illegal operations." (see Veditz, Summary, col. 3, lines 23-37).

In other words, the Veditz's LDID references language driver values, such as lookup tables, and does not correspond to any particular display screen. Accordingly, Veditz's LDID is not an identifier that corresponds with any "display strings," let alone any display strings retrieved from a file.

The Office Action contends that Applicants' next limitation, "writing the identifiers and the corresponding display strings to a translation template," is also taught by Veditz, citing col. 18, lines 10-67. This section of Veditz teaches basic terms and technology used in language configuration tables and known to those skilled in the art.

Veditz does not teach, however, writing the identifier (mistakenly identified as the "LDID" in the Office Action) along with a corresponding display string. It appears that the Examiner assumed that writing the display string and identifier to a template was synonymous with "automatically translating the display file into a format which is compatible with that currently employed by the system and setting the system language driver to one which is compatible with that of data file" as noted in the Office Action. This is a mistaken assumption. As described in detail above, Applicants' claimed invention writes the identifier and display string to a template file where it can be translated by automatic or human translators at a later time. In one embodiment, these translations are stored in separate files. That is why, in Applicants' invention, the identifiers are written back to the management definition data file. When a user uses the management definition data file, the identifiers stored in the management definition data file are used to locate the display strings in the user's native language.

This last limitation, "writing the identifier to the management definition data file," is also not taught or suggested by Veditz, as contended by the Office Action. Instead, Veditz teaches that various warning messages are displayed based on the LDID setting. The LDID however, does not correspond with a display string, however. As discussed above, Veditz makes it plainly clear how Veditz's LDID operates "for storing in desired location(s) of a data object information specifying the language driver that was in use when the data object was created or modified." In other words, language drivers may change over time and, as discussed in Veditz at col. 18, lines 10-67, the LDID helps in finding the language driver that was used for a particular data object. Importantly, however, Veditz never teaches or suggests "writing the identifier" to a file, as taught and claimed by Applicants.

As set forth above, Applicants have overcome the rejection of independent claims 1, 8, and 15 over Veditz. In addition, claims 2-7, 9-14, and 16-21 each depend, directly or indirectly, on one of these independent claims and each are each allowable for at least the same reasons that claims 1, 8, and 15 are allowable. Independent claims 21-23 are also allowable because they include the same limitations as claims 1, 8, and 15 along with additional limitations not claimed in claims 1, 8, and 15. Therefore, claims 21-23 are allowable for at least the same reasons as claims 1, 8, and 15. Finally, as discussed in the beginning of this Response, the Office Action never rejected claims 24 and 25. Applicants note that claims 24 and 25 also include the same limitations as set forth in claims 1, 8, and 15, and are therefore also allowable for at least the same reasons.

## Conclusion

As a result of the foregoing, it is asserted by Applicants that the remaining claims in the Application are in condition for allowance, and Applicants respectfully request an early allowance of such claims.

Applicants respectfully request that the Examiner contact the Applicants' attorney listed below if the Examiner believes that such a discussion would be helpful in resolving any remaining questions or issues related to this Application.

Respectfully submitted,

By /Joseph T. Van Leeuwen, Reg. No. 44,383/

Joseph T. Van Leeuwen, Reg. No. 44,383
Van Leeuwen & Van Leeuwen
Attorneys for Applicant
Telephone: (512) 301-6738
Facsimile: (512) 301-6742